**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**
**BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES**

| | | |
|---|---|---|
| In re Application of: Bauman et al. | § | |
| | § | Group Art Unit: 2143 |
| Serial No.: 10/038,008 | § | |
| Confirmation No.: 6602 | § | |
| | § | Examiner: Joseph E. Avellino |
| Filed: January 4, 2002 | § | |
| For: METHOD FOR DETERMINING ON DEMAND RIGHT SIZE BUFFERING WITHIN A SOCKET SERVER IMPLEMENTATION | § | |

MAIL STOP
APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

**CERTIFICATE OF MAILING OR TRANSMISSION**

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Mail Stop Appeal Brief - Patents, Commissioner for Patents, P. O. Box 1450, Alexandria, VA 22313-1450, or facsimile transmitted to the U.S. Patent and Trademark Office to fax number 571-273-8300 to the attention of Examiner Joseph E. Avellino, or electronically transmitted via EFS-Web, on the date shown below:

March 09, 2007 /Gero G. McClellan, Reg. No. 44,227/
Date Gero G. McClellan

## APPEAL BRIEF

Dear Sir:

Applicants submit this Appeal Brief to the Board of Patent Appeals and Interferences on appeal from the decision of the Examiner of Group Art Unit 2143 dated December 15, 2006, finally rejecting claims 1-7, 9-11, 14-15, 17-18, 20-22, 25, 26, 28-29. The final rejection of claims 1-7, 9-11, 14-15, 17-18, 20-22, 25, 26, 28-29 is appealed. This Appeal Brief is believed to be timely since it is facsimile transmitted by the due date of March 9, 2007, as set by the filing of a Notice of Appeal on January 9, 2007.

Please charge the fee of $500.00 for filing this brief to Deposit Account No. 09-0465/ROC920010193US2.

# TABLE OF CONTENTS

## Real Party in Interest

The present application has been assigned to International Business Machines Corporation, Armonk, New York.

## Related Appeals and Interferences

Applicant asserts that no other appeals or interferences are known to the Applicant, the Applicant's legal representative, or assignee which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

## Status of Claims

Claims 1-3, 5-7, 9-12, 14, 15, 17-23, 25, 26, 28 and 29 are pending in the application. Claims 1-29 were originally presented in the application. Claim 30 was added during prosecution. Claims 4, 8, 13, 16, 24, 27 and 30 have been canceled without prejudice. Claims 1-3, 5-7, 9-12, 14, 15, 17-23, 25, 26, 28 and 29 stand finally rejected. The final rejections of claims 1-3, 5-7, 9-12, 14, 15, 17-23, 25, 26, 28 and 29 are appealed. The pending claims are shown in the attached Claims Appendix.

## Status of Amendments

All claim amendments have been entered by the Examiner. No amendments to the claims were proposed after the final rejection.

## Summary of Claimed Subject Matter

Claimed embodiments include methods (see claims 1-3 and 5-7), computer programs stored on computer readable storage media (see claims 9-12, 14, 15, and 17-19) and computer systems (see claims 20-23, 25, 26, 28, and 29) directed to techniques for handling messages in a client-server environment. In particular, the computers of the client-server environment are sockets-based to facilitate a variety of I/O processing. *See Application*, 1:4-8, 9:5-8, *Abstract*. In one embodiment, a buffer is acquired only once client data has been received. Because the client data has already been received when the buffer for that data is acquired, the buffer may be sized exactly to the size of the client data, thereby making efficient use of storage. See *Application*, 6:29-32. For a description of the physical environment of the invention, see *Application*, p. 9-12, for a description of the software environment of the invention, see *Application*, p. 12-13, and for a description of right size buffering, see *Application*, p. 17-21.

CLAIM 1 – INDEPENDENT

Claim 1 recites a method of providing a logical framework for defining abstract A method of processing messages. See *Application*, 6:29-32, 6:34 - 7:1, 9:5-8. As claimed, the method includes receiving, at a socket configured for a server application executing on a computer, data from a remote source via a network connection prior to allocating a buffer to contain the data. See *Application*, 17:10-17, 19:15-20, Figure 10, 1012, 20:15-17, Figure 11, 1112. And subsequently, determining a mode to obtain the buffer according to a buffer mode parameter supplied with a receive operation call, wherein the buffer mode parameter indicates a buffer acquisition method for acquiring a buffer to contain the data received from a remote source via the network connection. See *Application*, 17:11-16. Several different buffer mode parameter types are discussed in the specification. For example, a "caller_supplied_dynamic" is described at *Application*, 17:34 – 18:1-6 -32, and a "system_supplied" is described at *Application*, 20:4-32.

As claimed, the method also includes obtaining the buffer according to the buffer acquisition method, wherein the obtained buffer is sized exactly to the size of the data

received from the remote source. See *Application*, 19:20-25, 19:29-33, 20:20-24, and 21:1-3. As claimed, the method also includes allocating the obtained buffer to contain the data. See *Application*, 19:20-25, 19:29-33, 20:20-24, and 21:1-3.

CLAIM 9 – INDEPENDENT

Claim 9 recites a computer readable medium containing a program which, when executed by a computer, performs operations for processing messages. See *Application*, 6:29-32, 7:3-9, and 9:10-23. As claimed, the operations include processing an input operation issued from a sockets server application to a sockets layer of the computer. See *Application*, 19:7-20, 20:3-15. As claimed, the input operation is configured with a buffer mode parameter indicating to the sockets layer a buffer acquisition method for acquiring a buffer for containing data received from a remote source via a network connection. See *Application*, 17:11-16. Several different buffer mode parameter types are discussed in the specification. For example, a "caller_supplied_dynamic" is described at *Application*, 17:34 – 18:1-6 -32, and a "system_supplied" is described at *Application*, 20:4-32.

As claimed, the operations also include receiving the data from the remote source via the network connection; subsequently. See *Application*, 17:10-17, 19:15-20, Figure 10, 1012, 20:15-17, Figure 11, 1112. And also include obtaining the buffer according to the buffer acquisition method, wherein the obtained buffer is sized exactly to the size of the data received from the remote source. See *Application*, 19:20-25, 19:29-33, 20:20-24, and 21:1-3. As claimed, the operations also include allocating the obtained buffer. See *Application*, 19:20-25, 19:29-33, 20:20-24, and 21:1-3.

CLAIM 15 –DEPENDENT

Claim 15 further specifies the operations recited by claim 14. Claim 15 specifies that the buffer is allocated from one of storage owned by the sockets server application

and system-supplied storage not owned by the sockets server application. See *Application*, 12:7-11, Figure 3, 372 and 374.

CLAIM 20 – INDEPENDENT

Claim 20 is directed to a system in a distributed environment. See *Application*, 6:29-32, 9:5-8, 10:5-10. As claimed, the system includes a network interface configured to support a network connection with at least one other computer in the distributed environment. See *Application*, 10:28-35, Figure 3, 368. The system also includes a memory comprising a sockets server application, a socket in communication with the sockets server application and a protocol stack in communication with the socket, wherein the protocol stack is configured to transport messages between the network interface and the socket and a processor configured to perform operations for processing messages. See *Application*, 11:9-25, Figure 10, 350, 10:34-35, Figure 10, 369.

As claimed, the operations include processing an input operation issued from the sockets server application to the socket, wherein the input operation is configured with a buffer mode parameter indicating to the socket a buffer acquisition method for acquiring a buffer for containing data received from the at least one other computer. See *Application*, 19:7-20, 20:3-15, 17:11-16. Several different buffer mode parameter types are discussed in the specification. For example, a "caller_supplied_dynamic" is described at *Application*, 17:34 – 18:1-6 -32, and a "system_supplied" is described at *Application*, 20:4-32.

As claimed, the operation also includes receiving the data. See *Application*, 17:10-17, 19:15-20, Figure 10, 1012, 20:15-17, Figure 11, 1112. And also include, subsequently, obtaining the buffer according to the buffer acquisition method, wherein the obtained buffer is sized exactly to the size of the data received from the remote source See *Application*, 19:20-25, 19:29-33, 20:20-24, and 21:1-3. As claimed, the

operations also include allocating the obtained buffer. See *Application*, 19:20-25, 19:29-33, 20:20-24, and 21:1-3.

CLAIM 26 –DEPENDENT

Claim 26 further limits the system of claim 20. Claim 20 specifies that the system of claim 20, further comprises application-supplied storage owned by the sockets server application and system-supplied storage not owned by the sockets server application. See *Application*, 12:7-11, Figure 3, 372 and 374. Claim 26 further specifies allocating the buffer from application-supplied storage when the buffer mode parameter has a first value and allocating the buffer from system-supplied storage when the buffer mode parameter has a second value. See *Application*, 12:7-11, Figure 3, 372 and 374.

## Grounds of Rejection to be Reviewed on Appeal

1. Rejection of claims 1-3, 5-7, 9-11, 14, 15, 17, 18, 20-22, 25, 26, 28 and 29 under 35 U.S.C. § 103(a) as being unpatentable over *Nair* (U.S. 2003/0217184) in view of *Beighe* (U.S. 6,055,576) in view of *Putcha* (U.S. 6,822,966).

2. Rejection of claims 12 and 23 under 35 U.S.C. § 103(a) as being unpatentable over *Nair* in view of *Beighe* and *Glassier* et al (U.S. 5,764,890).

3. Rejection of claim 19 under 35 U.S.C. § 103(a) as being unpatentable over *Nair* in view of *Beighe* in view of *Fry* (U.S. 4,467,411).

4. Rejection of claims 1-3, 5-7, 9-12, 14-15, 17-23, 25-26 and 28-29 under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claims 1-34 of co-pending Application No. 10/037,595.

## ARGUMENTS

### Claims 1-3, 5-7, 9-11, 14, 15, 17, 18, 20-22, 25, 26, 28 and 29 are patentable over *Nair* in view of *Beighe* and *Putcha*

*The Applicable Standard for Establishing Prima Facie Obviousness*

The Examiner bears the initial burden of establishing a *prima facie* case of obviousness. *See* MPEP § 2142. To establish a *prima facie* case of obviousness three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one ordinary skill in the art to modify the reference or to combine the reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. *See* MPEP § 2143. The present rejection fails to establish at least the first and third criteria.

Applicants submit that *Nair* fails to disclose a method performed by a server application in processing messages that includes receiving, at a socket configured for a server application executing on a computer, data from a remote source via a network connection prior to allocating a buffer to contain the data, as recited by claim 1. Claims 9 and 20 recite a similar limitation.

*Nair* is directed to a method of enhancing a data communications pathway by maintaining a pool of buffers used by a set of network communication protocol modules. *See Nair*, ¶ 25. *Nair* discloses that the protocol modules may share a pointer to a buffer across different layers of a protocol stack (e.g., passing a pointer from a physical layer, to a data link layer, to a network layer, etc.). Thus, *Nair* discloses maintaining the data frames in a common buffer space, referenced by a software module at each layer by the shared pointer. For example, *Nair* provides:

> In particular, when a frame or cell of data is received from a network attached to the machine, or when a packet of data is prepared for transmission over a network attached to the machine, as the data is processed by each appropriate protocol software module, the data is maintained in the same buffer space. Only the pointers to the data space need be passed between the protocol software modules so that the protocol software modules that process the data know where to access the data.

*Nair*, ¶ 20. Importantly, *Nair* teaches that the shared buffer used by the protocol stack may be discarded (or returned to the buffer pool) once a frame is provided to a server application. Specifically, *Nair* provides:

> "[P]rocessing of the data frame continues up the protocol stack until processing of the data frame by the machine is competed. **At such time, the data is read from the buffer at 230 and, for example, provided to an application software program.** At this point, for example, **the buffer is no longer needed** for temporarily storing the data pockets while the various protocol software modules in the protocol stack process the data frame."

*Nair*, ¶ 28. The "protocol software modules" discussed in *Nair* are include the TCP (and lower) layers of a TCP/IP stack and further, *Nair* expressly distinguishes the operations of these "protocol software modules" from those of a "higher level application." The following passages from *Nair* further illustrate this:

> [A] hardware interface, typically implemented in a chipset, provides a physical connection to the network. A driver, such as ATM driver 105, transmits and receives information, generally in the form of a well defined stream of binary digits, respectively to and from the hardware interface 103. The driver provides a mechanism to transmit and receive the stream of binary digits as a block of data, whether defined as a fixed length cell, as in the case of an ATM stream of data, or a variable length frame of data, as in the case of an Ethernet-based frame of data transmitted over a local area network. An Ethernet/IEEE 802.3 hardware interface, not shown, provides a physical connection to local area network 101 and essentially operates in the same manner to generally perform the same functions as ATM hardware interface 103.
>
> The ATM driver services higher layer protocol software modules in protocol stack implemented in the machine, such as PPP over ATM adaptation layer 5 (PPP over AAL5) software module 107 and Point to Point Protocol (PPP) software module 109. These modules, in turn, service, for example, a higher layer protocol software module such as IP software module 110. Likewise, the Ethernet driver services the IP software module 110. Finally, IP software module 110 services TCP software module 112.

*Nair*, ¶ 18-19. These passages describe elements of *Nair*, Figure 1, which illustrate a physical layer (H/W I/F 103), a data link layer (Ethernet 108 or ATM driver 105 and PPP over AALS 109), a network layer (IP 110), and a transport layer (TCP 11) of a network protocol stack. Figure 1 does not even illustrate the "higher level application;" instead, this figure simply includes an arrow leading from the TCP module 112.

Clearly, the operations performed by the server application are distinct from those used to manage a buffer within different layers of the protocol stack. The present claims, however, are directed to actions of a "server application" in processing data after it has been received over a socket connection, i.e., after the data is, in the words of *Nair*, "provided to an application software program." And claim 1 specifically recites a step of "receiving, at a socket configured for a server application ... data from a remote source via a network connection." Similarly, claims 9 and 20 recite a step of processing an input operation issued from a sockets server application ... [and] receiving the data from the remote source via the network connection. In each case, the claims are clearly directed to the operations of a server application, and not operations performed within layers of a protocol stack. The Examiner appears to agree with this point:

> Although Applicant's cited passages of the reference are correct, they do not have any bearing as to how *Nair* teaches the claimed limitations.

*Final Office Action*, p. 8. However, the Examiner goes on to suggest that:

> Applicants must see that "as an initial step, a driver or physical layer protocol software module [read server software application] receives a frame of data" (p. 3, ¶ 23). Then "the driver processes the data frame" (p. 3, ¶ 23). The allocation of the buffer occurs after the frame is received.

*Final Office Action*, p. 8. Frankly, the Examiner's suggestion that the "driver or physical layer protocol software module" is the same as the "server software application" makes no sense. *Nair* expressly distinguishes the driver (corresponding to a physical layer protocol module) from the "higher level application." Further, *Nair* expressly distinguishes the data link, network, and transport layers from the "higher level application." The distinction between the operations of the "software protocol modules" and those of the "higher level application" is readily apparent from *Nair's* discussion of processing of inbound data frames. Again:

> "[P]rocessing of the data frame continues up the protocol stack until processing of the data frame by the machine is competed. **At such time, the data is read from the buffer at 230 and, for example, provided to an application software program.** At this point, for example, **the buffer is no longer needed** for temporarily storing the data pockets while the various protocol software modules in the protocol stack process the data frame."

*Nair*, ¶ 28. Accordingly, based on the foregoing, Applicants submit that *Nair* does not disclose a method performed by a server application in processing messages, as recited by claims 1, 9, and 20.

Furthermore, Applicants submit that *Nair*, *Beighe,* and *Putcha* do not teach or suggest the recited step of "determining a mode to obtain the buffer according to a buffer mode parameter supplied with a receive operation call." The Examiner concedes that *Nair* and *Beighe* alone do not disclose this limitation, but suggests that the following passage from *Putcha* does:

> In one aspect, a network communication device for directing data units over a communication network includes at least one input and/or output port arranged to receive and transmit data units, a plurality of buffer units divided into several sub-pools, and a buffer allocator for allocating buffer units between the sub-pools. The buffer allocator is arranged to determine a priority value for each sub-pool based on quality of service parameter for each connection established at at least one input port. The buffer allocator is also arranged to determine a utilization value of the input port, and arranged to allocate buffer units for each sub-pool based on the priority value and based on the utilization value, wherein a minimal number of connections established at a most utilized port will suffer loss of data units while receiving the data units.

*Putcha*, 4:18-33. *Putcha* discloses "a network communication device" that includes a "buffer allocator for allocating buffer units between the sub-pools." *Putcha*, 4:18-24. The buffer allocator "is arranged to determine a priority value" and "to determine a utilization value." Thus, the buffer allocator is allocating the available buffer space based on priority and utilization values. Applicants respectfully submit that allocating buffers based on priority and utilization does not disclose determining a mode to obtain the buffer based on a buffer mode parameter supplied with a receive operation call. The buffer allocator of *Putcha* simply does not specify a mode of buffer acquisition; instead, buffers are allocated in a single/fixed manner that does not depend on the pool the buffer is acquired from.

Thus, although the Examiner asserts that:

> Applicant has not sufficiently defined what is meant by a 'buffer mode parameter' in the claim and is therefore open to interpretation. By this rationale, *Putcha* does, in fact, disclose a "buffer mode parameter" which

> could be construed as the priority value for the sub pool of *Putcha*. By this rationale, the rejection is maintained.

*Final Office Action*, p. 8, *Advisory Action*, Continuation sheet. However, claim 1 expressly recites that the "buffer mode parameter" is "supplied with a receive operation call" issued by the server application and further, that the "buffer mode parameter" "indicates a buffer acquisition method for acquiring a buffer to contain the data received from a remote source via the network connection." The Examiner's rationale that "*Putcha* does, in fact, disclose a 'buffer mode parameter' which could be construed as the priority value for the sub pool of Putcha" ignores the express limitations recited by claim 1 which specify that the "buffer mode parameter" is "supplied with a receive operation call," etc.

Applicants' argument is even more forceful when considering the additional limitations recited by claims 15 and 26. These dependent claims further specify that the buffer is allocated from "storage owned by the sockets server application" or from "system-supplied storage not owned by the sockets server application." In both cases, the claim further characterizes the actions of the "sockets server application" and not those of a "software protocol modules." Nevertheless, the Examiner suggests:

> *Beighe* further discloses that buffer is allocated from storage owned by the sockets server application based on a value of the buffer mode parameter (i.e. direction) (col. 3, lines 10-50)

*See Final Office Action*, p. 4. First of all, in regards to claims 1, 9, and 20, the Examiner concedes that "*Nair* in view of *Beighe* do not specifically disclose determine a buffer acquisition mode," *Final Office Action*, p. 4. Respectfully, it is completely contradictory, therefore, for the Examiner to assert that *Beighe* discloses that a buffer is allocated from storage owned by the sockets server application based on a value of the buffer mode parameter.

Moreover, the inherent "direction" in which data communication occurs is not the same as a "buffer mode parameter" that specifies to obtain a buffer from "storage owned by the sockets server application" or from "system-supplied storage not owned by the sockets server application." The cited passages from *Beighe* 3:10-50, describe the operations of a cable modem storing data packets "in a buffer in memory system

28." *Beighe*, 3:9-11. Even when combined with *Beighe*, it makes no sense to suggest that the "higher level application" from *Nair* or the "sockets server application" recited by the present claims would have access to the "memory system 28" of a cable modem. Finally, there is no teaching or suggestion in the cited material that a buffer may be allocated from different memory storage, i.e., from "storage owned by the sockets server application" or from "system-supplied storage not owned by the sockets server application," and instead, the only memory storage is "memory system 28," without the distinctions recited by claims 15 and 26.

For all the foregoing reasons, Applicants submit that claims 1-3, 5-7, 9-11, 14, 15, 17, 18, 20-22, 25, 26, 28 and 29 are patentable over *Nair*, in view of *Beighe*, and in further view of *Puchta*. Accordingly, Applicants respectfully request that this rejection be withdrawn.

**Claims 12 and 23 are patentable over *Nair* in view of *Beighe* in view of *Glassier***

First, on its face the rejection is defective. In a Response to the Final Office Action, Dated august, 9 2006, Applicants pointed out that claim 12 depends from claim 9, and claim 23 depends from claim 20, thus these claims include all of the limitations of claim 9 and 20, respectively. The Examiner concedes that *Nair* in view of *Beighe* does not disclose all the limitations recited by independent claim 9 and 20. As written, therefore, the rejection must fail.

Nevertheless, Applicants believe that the above discussion regarding claims 9 and 20 demonstrate that these claims are patentable over *Nair* in view of *Beighe* and *Putcha*. Thus, Applicants believe a detailed discussion of *Glasser* cited in regards to dependent claims 12 and 23 is unnecessary.

**Claim 19 is patentable over *Nair* in view of *Beighe* in view of *Fry***

Like the rejection of claims 12 and 23, the rejection is defective on its face. As pointed out in Applicant's Response to the *Final Office Action*, Dated august, 9 2006,

Claim 19 depends from claim 9, and thus includes all of the limitations recited by this claim. The Examiner concedes that *Nair* in view of *Beighe* does not disclose all the limitations of independent claim 9. Thus, when the Examiner asserts "[r]eferring to claim 19, *Nair* in view of *Beighe* discloses the invention substantively as described in claim 9," presumably, the Examiner believes *Nair* in view of *Beighe* does not disclose each limitation recited by claim 9. As written, therefore, the rejection must fail.

Nevertheless, Applicants believe that the above discussion regarding claims 1, 9, and 20 demonstrate that these claims are patentable over *Nair* in view of *Beighe* and *Puchta*. Thus, Applicants believe a detailed discussion of the *Fry* reference cited in regards to dependent claim 19 is unnecessary.

**Claims 1-3, 5-7, 9-12, 14-15, 17-23, 25-26 and 28-29 and claims 1-34 of co-pending Application No. 10/037,595**

Claims 1-3, 5-7, 9-12, 14-15, 17-23, 25-26 and 28-29 are rejected under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claims 1-34 of co-pending Application No. 10/037,595.

Applicants acknowledge the double patenting rejection made in the Final Office Action mailed August 9, 2006, and respectfully request that the rejection be held in abeyance because (i) no claim in the present application is currently allowable and (ii) the application on which the rejection is made (No. 10/037,595) has not issued. Because it is possible that no claims will issue, or that the claims of the present application will be amended in such a way to overcome the Examiner's concerns regarding double patenting, Applicants defer responding until the present rejection ripens into an actual double patenting rejection.

## CONCLUSION

The Examiner errs in finding that:

- Claims 1-3, 5-7, 9-12, 14, 15, 17-23, 25, 26, 28 and 29 are unpatentable over *Nair* in view of *Beighe* and *Putcha*;
- Claims 12 and 23 are unpatentable over *Nair* in view of *Beighe* in view of *Glassier*;
- Claim 19 under 35 U.S.C. § 103(a) is unpatentable over Nair in view of *Beighe* in view of *Fry*; and
- Claims 1-3, 5-7, 9-12, 14-15, 17-23, 25-26 and 28-29 are patentable over claims 1-34 of co-pending Application No. 10/037,595.

Withdrawal of the rejections and allowance of all claims is respectfully requested.

Respectfully submitted, and
S-signed pursuant to 37 CFR 1.4,

/Gero G. McClellan, Reg. No. 44,227/

Gero G. McClellan
Registration No. 44,227
Patterson & Sheridan, L.L.P.
3040 Post Oak Blvd. Suite 1500
Houston, TX 77056
Telephone: (713) 623-4844
Facsimile: (713) 623-4846
Attorney for Appellant(s)

## CLAIMS APPENDIX

1. (Previously Presented) A method of processing messages, comprising:

receiving, at a socket configured for a server application executing on a computer, data from a remote source via a network connection prior to allocating a buffer to contain the data; and subsequently:

determining a mode to obtain the buffer according to a buffer mode parameter supplied with a receive operation call, wherein the buffer mode parameter indicates a buffer acquisition method for acquiring a buffer to contain the data received from a remote source via the network connection;

obtaining the buffer according to the buffer acquisition method, wherein the obtained buffer is sized exactly to the size of the data received from the remote source; and

allocating the obtained buffer to contain the data.

2. (Original) The method of claim 1, wherein the messages are client-server messages.

3. (Original) The method of claim 1, wherein the data is received over a sockets streaming protocol.

4. (Canceled)

5. (Previously Presented) The method of claim 1, wherein the allocating is performed in response to a buffer request from the socket.

6. (Previously Amended) The method of claim 1, wherein the network connection is a Transport Control Protocol/Internet Protocol (TCP/IP) connection.

7. (Original) The method of claim 1, wherein allocating the buffer comprises:

processing a buffer request from a sockets layer after receiving the data; and

providing the buffer to the sockets layer.

8. (Canceled)

9. (Previously Presented) A tangible computer readable medium containing a program which, when executed by a computer, performs operations for processing messages, the operations comprising:

processing an input operation issued from a sockets server application to a sockets layer of the computer, wherein the input operation is configured with a buffer mode parameter indicating to the sockets layer a buffer acquisition method for acquiring a buffer for containing data received from a remote source via a network connection; receiving the data from the remote source via the network connection;

subsequently obtaining the buffer according to the buffer acquisition method, wherein the obtained buffer is sized exactly to the size of the data received from the remote source; and

allocating the obtained buffer.

10. (Previously Presented) The tangible computer readable medium of claim 9, wherein the messages are client-server messages.

11. (Previously Presented) The tangible computer readable medium of claim 9, wherein the data is received over a sockets streaming protocol.

12. (Previously Presented) The tangible computer readable medium of claim 9, wherein the input operation is further configured with a record definition specifying to the sockets layer a format of the data.

13. (Canceled)

14. (Previously Presented) The tangible computer readable medium of claim 10, wherein the allocation is performed by one of the sockets server application and the sockets layer.

15. (Previously Presented) The tangible computer readable medium of claim 10, wherein the buffer is allocated from one of:

storage owned by the sockets server application; and

system-supplied storage not owned by the sockets server application.

16. (Canceled)

17. (Previously Presented) The computer tangible readable medium of claim 10, wherein allocating the buffer comprises executing a callback function provided by the sockets server application with an instruction to allocate the buffer.

18. (Previously Presented) The tangible computer readable medium of claim 10, wherein the allocating is performed in response to a buffer request made by the sockets layer.

19. (Previously Presented) The tangible computer readable medium of claim 9, further comprising:

if the buffer is large enough to contain the data, copying the data into a previously allocated buffer provided to the sockets layer with the input operation; and

if the previously allocated buffer is not large enough to contain the data, requesting a larger buffer sufficient to contain the data in accordance with the buffer acquisition method.

20. (Previously Presented) A system in a distributed environment, comprising:

a network interface configured to support a network connection with at least one other computer in the distributed environment;

a memory comprising a sockets server application, a socket in communication with the sockets server application and a protocol stack in communication with the socket, wherein the protocol stack is configured to transport messages between the network interface and the socket;

a processor configured to perform operations for processing messages, the operations comprising:

processing an input operation issued from the sockets server application to the socket, wherein the input operation is configured with a buffer mode parameter indicating to the socket a buffer acquisition method for acquiring a buffer for containing data received from the at least one other computer; and

receiving the data; subsequently

obtaining the buffer according to the buffer acquisition method, wherein the obtained buffer is sized exactly to the size of the data received from the remote source; and

allocating the obtained buffer.

21. (Original) The system of claim 20, wherein the messages are client-server messages.

22. (Original) The system of claim 20, wherein the protocol stack is configured for a sockets streaming protocol.

23. (Original) The system of claim 20, wherein the memory comprises record definition specifying to the socket a format of the data.

24. (Canceled)

25. (Previously Presented) The system of claim 20, wherein the allocation is performed by one of the sockets server application and the socket.

26. (Previously Presented) The system of claim 20, further comprising application-supplied storage owned by the sockets server application and system-supplied storage not owned by the sockets server application and wherein allocating the buffer is dependent on a value of the buffer mode parameter and comprises one of:

allocating the buffer from application-supplied storage when the buffer mode parameter has a first value; and

allocating the buffer from system-supplied storage when the buffer mode parameter has a second value.

27. (Canceled)

28. (Previously Presented) The system of claim 20, wherein allocating the buffer comprises executing a callback function provided by the sockets server application with an instruction to allocate the buffer.

29. (Previously Presented) The system of claim 20, wherein the allocating is performed in response to a buffer request made by the socket.

30. (Canceled)

## EVIDENCE APPENDIX

None.

## RELATED PROCEEDINGS APPENDIX

None.